Vorwort

In den letzten zwanzig Jahren habe ich viele Bücher über Softwareentwicklung gelesen. Die meisten von ihnen enthielten Informationen, die mir auch anderweitig zugänglich waren. In der Vergangenheit kam dieses Material häufig aus Magazinen, in letzter Zeit hingegen aus dem Internet. Einige der Bücher basierten jedoch stattdessen auf der praktischen Erfahrung des Autors in der Kunst und Wissenschaft der Softwareentwicklung.

Heute kann ein Autor, der noch nie ein Gericht gekocht hat, ein Buch darüber schreiben, wie man zum Meisterkoch wird. Er oder sie kann einfach eine Sammlung von Rezepten aus dem Internet herunterladen, umformulieren und neu anordnen – und siehe da: Ein neues Kochbuch kommt auf den Markt! Besonders auf dem Gebiet der Softwareentwicklung werden wir mit Büchern überflutet, die auf diese Weise zu Stande kommen.

Als man mich bat, ein Buch über XML, Java und JSP zu schreiben, bestand ich darauf, dass es ein praktisches Anleitungshandbuch sein müsse, das auf wirklicher Entwicklungserfahrung beruht. Die Zielgruppe des Buchs würden Softwareentwickler sein, die diese Technologien zu verstehen und in realen Webanwendungen einzusetzen versuchen. Ich wusste, dass ich mir selbst »die Kochschürze umbinden« müsste, um für diese Leser ein Buch zu schreiben. Der kritische Teil würde eine funktionale, relevante Webanwendung mit plausibler Nützlichkeit sein.

Ich begann mit Recherchen im Internet und hatte schon bald sehr viele relevante Informationen zusammengetragen. Ich legte auch fest, welche der zahlreichen verfügbaren Tools und Produkte für das Projekt dieses Buchs Verwendung finden würden. Die meisten davon befanden sich im Zustand schneller Entwicklung. Entwicklung und Implementierung von Standards und Empfehlungen für diese modernsten Technologien bedürfen einer weltweiten Anstrengung.

Selbst für einen erfahrenen Softwareentwickler war es keine einfache Aufgabe, diese Informationen und Technologien in ein praktisches Beispiel einzubeziehen. Zahlreiche gut dokumentierte Tools waren veraltet, während aktuellere Tools häufig unzureichend dokumentiert waren. Äußerst aktive Mailinglisten wurden von Entwicklern frequentiert, die solche Tools und Standards bereits früh für reale Webanwendungen übernommen haben. Diese Pioniere zeigten mir schon bald, dass zwischen den Tools und den Standards viele Inkompatibilitäten bestehen.

Der anfängliche Plan bestand darin, in den ersten Kapiteln die Tools und Technologien vorzustellen und anschließend das Projekt, das Beispiel einer Webanwendung, durchzuführen. Es wurde mir klar, dass dies bedeutet hätte, »das Pferd von

hinten aufzuzäumen«. Ich entschied, dass meine erste Aufgabe in der Erstellung der Beispielanwendung bestehen würde und dass ich erst dann – im Kontext dieses Projekts – darüber schreiben würde, wie XML- und Java-basierte Technologien von Lesern eingesetzt werden können.

Damit begann das Projekt Spaß zu machen. Ich entwarf und implementierte eine Chat-Webanwendung mit dem Namen »bonForum«. Sie basiert auf XML, Java-Servlets und -Applets und auf JSP. Sie stellte mir die herausfordernsten Aufgaben des Webanwendungsdesigns. Da es sich um eine sehr beliebte und relevante Webanwendung handelt, hoffe ich, dass sie Leser interessieren wird. Als Experiment und Lernprogramm sind ihr Design und ihre Implementierung bei weitem nicht perfekt.

Über Fehlerberichte, Korrekturvorschläge, generelles Feedback und allgemeine Kommentare freue ich mich. Über email@www.bonforum.org können Sie sich mit mir in Verbindung setzen. Errata, Versionsaktualisierungen, Mailinglisten und verwandte Informationen finden Sie unter http://www.bonforum.org und http://www.galileocomputing.de.

Westy Rockwell
Software Developer
Tarent GmbH
Bonn, Germany





1 Einführung und Anforderungen

In diesem Kapitel erfahren Sie, welches Wissen wir Ihnen mit diesem Buch vermitteln wollen. Außerdem begründen wir unsere Entscheidungen bezüglich der Art und Weise, in der die »praktische« Seite dieses Buchs unterstützt wird. Wir begründen auch die Auswahl der Software-Tools und -Bibliotheken, die wir zur Veranschaulichung eines umfangreichen Themas ausgewählt haben: der Entwicklung von XML- und Java-basierten Webanwendungen.

1.1 Das Ziel dieses Buchs

Das Ziel dieses Buchs besteht, kurz gesagt, darin, einen einigermaßen erfahrenen Softwareentwickler in die Lage zu versetzen, Webanwendungen basiert auf XML, XSLT, Java-Servlets und JavaServer Pages zu schreiben. Darüber hinaus werden wir versuchen, dieses Ziel mit der günstigsten und unkompliziertesten Hardware und Software zu erreichen.

Dieses Buch ist nicht als umfassende Referenz für XML, XSLT, Java-Servlets oder JSP gedacht. Es ist auch nicht als Einführung für XML, XSLT, Java-Servlets oder JSP gedacht. Wenn es jedoch Ihr Ziel ist, zu einem Entwickler von Webanwendungen zu werden, und wenn Sie noch keine Erfahrung mit XML-Technologie oder serverseitiger Java-Technologie haben, sollten Sie mit dem Kapitel »Online-Informationsquellen« beginnen. Dort werden Sie auch einige Links finden, die Ihnen helfen, alle Informationen zu finden, die Sie benötigen, um dieses Buch zu verstehen.

1.2 Zur Verwendung dieses Buchs

Es gibt viele Bücher, die man sehr gut auf eine längere Flugreise mitnehmen kann. Dieses Buch gehört wahrscheinlich nicht dazu, es sei denn, Sie haben einen sehr modernen Laptop auf die Reise mitgenommen! Wir sind der Auffassung, dass es nicht möglich ist, etwas über die XML-Programmierung mit Java zu lernen, ohne im Internet zu surfen. Dieses Buch wird am besten teils als »Laboranleitung« und teils als »Reiseführer« angesehen, dessen Nützlichkeit davon abhängt, wie häufig Sie die angegebenen Webadressen tatsächlich besuchen.

Wir sind uns nur allzu schmerzlich der Tatsache bewusst, dass wir heute im Internet zum Thema des Buchs »Tatsacheninformationen« finden können, die zu dem Zeitpunkt seines Erscheinens bereits »Fiktionen« oder zumindest irrelevant sein werden. Der Grund hierfür ist identisch mit demjenigen, der das Thema so aktuell macht: Die zunehmende Relevanz und Popularität von XML hat seine rasante Entwicklung zur Folge. Wir können also nur hoffen, dass dieses Buch Ihnen als Starthilfe bei einer langen (und geduldigen?) Reise im Cyberspace dient.

XML- und Java-basierte Softwaretechnologien sind nicht besonders neu. In dem Maße, in dem man sich ihrer Leistungsmöglichkeiten und ihres Nutzens bewusst wird, nimmt jedoch auch ihre Popularität zu. Besonders aktiv sind die zahlreichen Mailinglisten zum Thema XML, JSP und Java Servlets, in denen weltweit Tausende von Entwicklern in einer angeregten Diskussion und in lebhaftem Informationsaustausch stehen. Wenn Sie unseren Rat befolgen und einige dieser Mailinglisten abonnieren, werden Sie schon bald selbst Zeuge dieser rasanten Popularitätszunahme und der wachsenden Geschwindigkeit, mit der sich XML und Java weiterentwickeln, denn Ihre Mailbox wird sich täglich mit Hunderten von Nachrichten füllen.

Einige Anmerkungen über den Begriff »JSP«

Wir fanden unterschiedliche Meinungen darüber, wie der Begriff »JSP« aufzufassen sei, und deshalb ist es wahrscheinlich eine gute Idee zu erklären, wie wir ihn in diesem Buch verwenden. Der Name »JavaServer Pages« ist der Name, den Sun einer Technologie gegeben hat. Er ist äquivalent zu seiner Abkürzung, die »JSP« lautet. Sowohl »JavaServer Pages« als auch »JSP« sind geschützte Markennamen. Aus diesen Gegebenheiten folgern wir einige Regeln:

1. Es ist nicht korrekt, den Namen »Java Server Pages« anstelle von »JavaServer Pages« für das Produkt von Sun zu benutzen.
2. Es ist nicht korrekt, den Ausdruck »JavaServer Page« für eine Seite (Datei oder Dokument) zu benutzen.
3. Es ist nicht korrekt, den Ausdruck »JSP« für eine Seite (Datei oder Dokument) zu benutzen.
4. Es ist nicht korrekt, den Ausdruck »JSPs« für mehrere Seiten (Dateien oder Dokumente) zu benutzen.
5. Es ist korrekt, die Ausdrücke »JSP Seite« und »JSP Seiten« zu benutzen.
6. So seltsam es klingt, ist es doch korrekt, den Ausdruck »JavaServer Pages Seite(n)« zu benutzen.

Wir fanden sämtliche oben beschriebenen Varianten im Netz. Wir versuchen uns an die oben genannten Regeln zu halten und hoffen, den Begriff in einheitlicher, an Sun orientierter Weise zu verwenden.

1.3 Einige verwirrende XML-Informationen im Internet

Während dieses Popularitätszuwachses haben insbesondere auch die mit XML in Zusammenhang stehenden Technologien Änderungen erfahren und sind mit großem Tempo angewachsen. Dies hat zu einiger Verwirrung bezüglich der Informationen geführt, die Sie im Internet finden werden. So gibt es beispielsweise zahlreiche Beiträge zu Mailinglisten, die nützliche Tipps oder nützlichen Code enthalten, ohne jedoch anzugeben, welche Server, Browser und Tools und welche Versionen von dem Entwickler verwendet wurden, der diese Beiträge an die Mailingliste geschickt hat.

Die vielleicht größte Quelle der Verwirrung für den Neuling auf dem Gebiet webbasierter XML-Informationen besteht darin, dass der am weitesten verbreitete und fortschrittlichste Webbrowser (Microsoft Internet Explorer) sich entschieden hat, einige mit XML in Zusammenhang stehende Vorschläge in bestimmte »inoffizielle« Richtungen zu erweitern. Es lässt sich argumentieren, dass dies erforderlich war, um diese XML-Vorschläge zur damaligen Zeit verwenden zu können. Obwohl auf diese Weise ein nützlicherer Browser entstanden ist, hat dies jedoch dazu geführt, dass sich stark durchsetzende »Dialekte« dieser Vorschläge entstanden sind – Dialekte, die von den »eigentlichen« Standards für XML-Technologien, die sich erst später entwickelt haben, beträchtlich abweichen.

In jüngster Zeit hat die Version 5.5 des Internet Explorers einen Teil dieser »eigentlichen« Standards übernommen, die von der mit der Wahrung dieser Standards beauftragten Institution, dem World Wide Web Consortium oder W3C (http://www.w3.org/), empfohlen werden. Microsoft beabsichtigt offenbar, die W3C-Empfehlungen in einer künftigen Version des Internet Explorers vollständig zu implementieren. Zwischenzeitlich hat das Unternehmen jedoch zur Zunahme der Zahl herstellerabhängiger Unterschiede in XML-basierten Technologien beigetragen.

1.4 Entscheidungen, mit denen Entwickler von XML- und Java-Webanwendungen konfrontiert sind

Wenn Sie sich entscheiden, als Entwickler XML und Java zu verwenden, sind Sie sogleich mit einem Dilemma konfrontiert. Sollen Sie die Vorteile der XML-basierten Programmierung nutzen, die in den meisten Windows- und NT-Computern zur Verfügung steht? Oder sollen Sie stattdessen nur verwenden, was den Standards entspricht? Und wenn das Letztere – welche Version dieser Standards? Und welche Tools, Entwicklungsumgebungen und Programmiersprachen?

1.4.1 Verarbeitung auf der Client- oder Serverseite

Die Nutzung der XML-Technologien von Microsoft bedeutet in der Regel eine Übernahme der Verarbeitung auf die Clientseite, da die DLLs mit dem Webbrowser installiert werden. Viele Entwickler, die sich für diesen Weg entscheiden, werden versucht sein, mithilfe von Microsoft-Entwickler-Tools zu arbeiten, besonders mit Visual Basic. Praktische Informationen zu dieser Vorgehensweise können Sie dem Begleitbuch zu dieser Publikation, **XML mit VB und ASP** von Elmar Geese und Markus Heiliger, entnehmen.

Die Verarbeitung in Webanwendungen auf der Serverseite durchzuführen, bietet jedoch zahlreiche Vorteile. Einer dieser Vorteile ist besonders zwingend für Entwickler, zu deren Aufgaben es in der Vergangenheit gehörte, in größeren Netzwerken verteilte Software zu unterstützen, die auf jedem Client installiert und konfiguriert werden musste! Eine Webanwendung, die geändert, angepasst und entwickelt werden kann, indem lediglich die Software auf einigen Servern überarbeitet wird, lässt sich wesentlich einfacher installieren, warten und unterstützen als eine Webanwendung, die auf Programmen basiert, die auf Tausenden von Clients installiert und konfiguriert werden müssen.

Java-Applets und andere herunterladbare Softwaremodule erlauben Ihnen natürlich den Einsatz clientseitiger Verarbeitung in Webanwendungen. Sie stellen jedoch unterschiedliche Anforderungen und zeigen in verschiedenen Webbrowsern ein unterschiedliches Verhalten. Dies bedeutet, dass Ihre Webanwendung zwar von den meisten, aber nicht von allen potenziellen Benutzern verwendet werden kann. Außerdem wird die Benutzerzufriedenheit bei Browsern von der Geschwindigkeit und der Bandbreite der Webverbindungen abhängen.

1.4.2 Welcher Webserver sollte verwendet werden?

Aus den genannten Gründen ziehen es zahlreiche Entwickler vor, bei der Erstellung von Webapplikationen die Verarbeitung dem Server zu überlassen. Aber auch bei dieser Vorgehensweise steht man vor einem Dilemma, das statt mit dem Browser in diesem Fall mit dem Webserver in Zusammenhang steht. Der Verkehr im Internet wird hauptsächlich über drei Typen von Webservern abgewickelt. Einer von diesen ist natürlich Microsofts Internet Information Server (IIS). Ein anderer ist Netscapes Enterprise Server. Der dritte ist der Apache Server, der nach einem neuerlichen Bericht in 57% aller Serverinstallationen auf dem Internet verwendet wird.

Die Frage stellt sich also nun folgendermaßen: Werden diese drei am häufigsten verwendeten Webserver als Host für die Webanwendungen dienen können? Wel-

che Implementierung der verschiedenen XML- und Java-Tools müssen wir wählen, um es einem bestimmten HTTP-Server zu ermöglichen als Host der Webanwendung dienen zu können? Oder sollten wir unsere Webanwendung einfach auf einem der zahlreichen verfügbaren kommerziellen Webanwendungs-Frameworks erstellen?

1.4.3 Welche Plattform(en) soll(en) verwendet werden?

Entwickler von Webanwendungen sehen sich anfänglich noch zahlreichen anderen Fragen gegenüber. Auf welcher Plattform (welchem Betriebssystem) sollen die Komponenten unserer Webanwendung auf der Serverseite ausgeführt werden? Ein Entwickler könnte sich für Microsoft Windows NT entscheiden, besonders dann, wenn er bereits mit den Entwicklungs-Tools und -umgebungen dieses Betriebssystems vertraut ist, oder wenn ein Internet Information Server (IIS) der Webserver der Anwendung sein wird. Java-Entwickler werden hingegen wahrscheinlich nach einer plattformunabhängigen Lösung suchen und diese Lösung dann auf der Plattform ihrer Wahl entwickeln, sei dies Windows NT, Linux, Solaris oder eine andere.

1.4.4 Welche Programmiersprache(n) sollte(n) verwendet werden?

Wenn es eine Computersprache gäbe, die nicht mehr versuchen würde, sich weiterzuentwickeln und für die Erstellung von Webanwendungen nützlich zu bleiben, dann wäre sie wahrscheinlich eine tote Sprache. Dem Entwickler, der Webanwendungen erstellen will, stehen zahlreiche verschiedene Programmiersprachen zur Auswahl. Einige der wichtigeren Sprachen sind Java, C++, Visual Basic, Perl und Python. Zudem stehen zahlreiche Bücher und andere Ressourcen zur Verfügung, die Ihnen bei der Arbeit mit diesen und anderen Sprachen helfen können.

Wie aus dem Titel dieses Buchs bereits hervorgeht, gehen wir davon aus, dass Ihren Entwicklungsbemühungen XML, XSLT, Java und JSP zu Grunde liegen wird. Als direkte Konsequenz hieraus ergibt sich, dass unsere eigenen Antworten auf alle diese Fragen stark in die Richtung bestimmter und nicht in die Richtung alternativer Technologien tendieren. Es ist allerdings nicht unsere Absicht mit unserer Wahl der Plattform, der Tools und der Softwarekomponenten zu implizieren, dass andere Möglichkeiten schlechter sind als diejenigen, für die wir uns hier entschieden haben.

1.5 Für dieses Buch getroffene Entwicklungsentscheidungen

Das vorliegende Buch erhebt keineswegs den Anspruch, sämtliche Antworten auf die obigen Fragen abzudecken. Wir wählen stattdessen einen Satz möglicher Antworten aus. Unsere Hoffnung ist es, dass Ihnen unsere Erfahrungen mit den von uns gewählten Tools und Komponenten wertvoll genug erscheinen, auch wenn Sie für sich selbst andere Antworten gefunden haben. Unsere strategischen Entscheidungen werden in den folgenden Abschnitten vorgestellt.

1.5.1 Unsere Entwicklungsplattform für die Beispiele

Basierend auf den für dieses Buch gewählten Zielen, gäbe es zahlreiche gute Gründe, als Entwicklungsplattform Linux zu wählen. Zum einen steht dieses Betriebssystem kostenlos zur Verfügung, und wir beabsichtigen, die Lernkosten so gering wie möglich zu halten. Zum anderen werden, aufgrund der Beliebtheit des Apache Servers, unter Linux wahrscheinlich weltweit mehr Webserver ausgeführt als unter jedem anderen Betriebssystem.

Wir werden unsere Beispiele jedoch auf einem Windows NT Server, Version 4.0 (Build 1381: Service Pack 5) entwickeln. Wir gehen dabei davon aus, dass bei dieser Wahl die meisten unserer Leser auf dieser Plattform entwickeln können. Wenn Sie nicht zu dieser Gruppe gehören, bitten wir Sie um Entschuldigung. Sie müssen in unseren Beispielen einiges ändern, bevor Sie sie auf Ihrer speziellen Plattform ausführen können.

- Die zur Angabe von Dateipfaden in diesem Buch verwendeten Zeichen entsprechen normalerweise der Windows-Vorgabe, wobei umgekehrte Schrägstriche (»\«) als Trennzeichen zwischen Ordnernamen stehen. Wenn Sie unter UNIX arbeiten, müssen Sie diese Trennzeichen durch einfache Schrägstriche (»/«) ersetzen.
- Wir werden in diesem Buch die Verwendung einiger Windows-Stapeldateien (.bat) erläutern. Wenn Sie auf einem Linux- oder Unix-System arbeiten, müssen Sie stattdessen die entsprechenden Shellskripte (.sh) verwenden.
- Die von uns verwendeten Softwarepfade gehen davon aus, dass Sie auf einem Windows-System arbeiten. Die entsprechenden Pfade für Linux- oder Unix-Systeme weichen von den angegebenen Pfaden ab.
- Alle von uns in dieses Buch aufgenommene Abbildungen stammen vom Bildschirm eines NT-Server-Systems. Wenn Sie unter Linux, Unix oder einer anderen Plattform arbeiten, müssen Sie dies berücksichtigen.

Wir werden versuchen, die Konsequenzen unserer Wahl der Entwicklungsplattform auf ein Minimum zu beschränken.

1.5.2 Unsere Java-Entwicklungsumgebung

Wir sind uns bewusst, dass Sie als Leser dieses Buchs bereits eine bevorzugte Java-Entwicklungsumgebung haben, vielleicht VisualAge für Java von IBM, JBuilder von Inprise/Borland oder irgendeine der zahlreichen anderen Umgebungen. Sie würden es sicherlich bevorzugen, wenn auch wir das Tool verwendeten, mit dem Sie arbeiten möchten.

Theoretisch sollte die Wahl der Entwicklungsumgebung keinen Einfluss auf die Java-Webanwendung haben, denn das ist schließlich die Zielsetzung von Java. Praktisch wird sie jedoch die Art beeinflussen, in der wir die Informationen darstellen werden. Dies wird eventuell auch Ihre Möglichkeit betreffen, den Beispielen zu folgen und sie tatsächlich zu kompilieren und auszuführen. Wir wählen daher die preiswerteste Entwicklungsumgebung, die von jedermann verwendet werden kann.

Es gibt natürlich keine wirklich kostenlose Software. Wenn Sie sie der zum Lieferumfang dieses Buchs gehörenden CD-ROM entnommen haben, haben Sie bereits dafür bezahlt! Und selbst wenn Sie sie aus dem Internet herunterladen, könnte sie Sie einiges an Verbindungsgebühren kosten. Außerdem benötigen Sie Hardware, um kostenlose Software auszuführen – möglicherweise werden einige unserer Leser Geld für ein Upgrade Ihrer Hardware ausgeben müssen, damit diese den von der Java-Entwicklung gestellten Anforderungen genügt.

Wir gehen davon aus, dass Sie zumindest die frei verfügbare Version des Sun JDK1.2.2 auf Ihrem NT-System installiert haben. Sie werden stattdessen wahrscheinlich auch mit der Version 1.3 arbeiten können, obwohl Sie möglicherweise feststellen werden, dass einiges anders funktioniert oder aussieht. Frühere Java-Versionen arbeiten mit der von uns verwendeten, mit XML in Zusammenhang stehenden Software jedoch nicht.

Wir hatten zunächst erwogen, Sie aufzufordern, den Beispielen dieses Buchs unter Verwendung der Windows NT-Befehlsfensterschnittstellen (bzw. Eingabeaufforderung unter Windows 95/98/2000) zum Java-Compiler und den dazugehörenden Tools zu folgen. Zahlreiche, dieses Buch lesende Java-Programmierer werden mit diesen Befehlszeilenoberflächen bereits vertraut sein, und andere könnten sicherlich davon profitieren, sie ebenfalls kennen zu lernen. Auf Fenstern basierende Entwicklungsumgebungen haben sich jedoch aus guten Gründen weiterentwickelt, und so haben wir uns stattdessen entschieden, ElixirIDE zu verwenden. Hierbei handelt es sich um eine Java-integrierte Entwicklungsumgebung, die – obwohl sie ein leistungsstarkes Tool darstellt – die Ressourcen des Lesers (an Hardware und Geld!) nicht allzu sehr strapaziert.

Es gibt noch andere integrierte Java-Entwicklungsumgebungen, die kostenlos verfügbar und sicherlich sehr gut sind. Nach unseren Erfahrungen erfordern sie jedoch mehr Ressourcen, insbesondere im Hinblick auf den Arbeitsspeicher des Systems, als die von uns gewählte Umgebung ElixirIDE.

1.5.3 Serverbasierte Webanwendungsarchitektur

Eine weitere der von uns diskutierten am Anfang stehenden Fragen war, ob bei der Entwicklung von Webapplikationen die Verarbeitung stärker auf der Client- oder auf der Serverseite erfolgen sollte. Wir sind der Auffassung, dass die Vorteile einer Implementierung mit verstärkter Verarbeitung auf der Serverseite die Nachteile dieser Alternative aufwiegen.

Ein wohl bekannter Vorteil der Verarbeitung im Webbrowser (auf der Clientseite) besteht darin, dass hierdurch der Netzwerkverkehr reduziert wird. Der Browser muss hierbei nicht ständig auf den Server zugreifen, um eine andere Anzeige für den Benutzer zu laden, da er hierzu auf die lokal verfügbare Webanwendung zugreifen kann. Wir meinen jedoch, dass es noch andere Möglichkeiten zur Reduzierung des Netzwerkverkehrs gibt, die die Verwendung von XML- und XSLT-Technologien involvieren. Wenn diese mit den Vorteilen der Verarbeitung auf der Serverseite kombiniert werden, sollten sie Sie davon überzeugen, sich für die Verarbeitung auf dem Webserver zu entscheiden.

> Den Beispielen in diesem Buch liegt eine klare Bevorzugung der Verarbeitung auf dem Server statt auf dem Client zu Grunde.

Um konsistent zu sein, müssten wir eigentlich bei der Wahl des Browsers neutral sein. Dies würde bedeuten, dass wir Ihnen »farblosen« HTML-Code für die Browser bieten, einen Code, der von den bevorzugten und weniger bevorzugten Browsern jeder Plattform interpretiert werden kann.

Da wir uns jedoch für NT Server als Betriebsystem entschieden haben, ist der Internet Explorer für uns kostenlos. Getreu unserem Ziel einer kostengünstigen Entwicklung werden wir in allen Beispielen Internet Explorer 5.0 (kurz: IE5.0) verwenden. Wir überlassen es Ihnen, die Beispiele auf anderen Browsern zu testen und die dafür an unseren Beispielen notwendigen Änderungen vorzunehmen.

Ein weiterer Grund für unsere Wahl von Internet Explorer als Browser besteht darin, dass es der im Internet am häufigsten verwendete Browser ist. Es ist daher wohl ratsam, unsere Entwicklungsbemühungen auf diesem Browser zuerst zu testen!

1.5.4 XML-Standards von W3C

Eine weitere Frage, die sich bei der Entwicklung von Webanwendungen stellt, ist die, welche XML-Standards befolgt werden sollen. Die Antwort hierauf fällt leicht: Befolgen wir die »echten« Standards. Dies sind die Standards, die von einer als »World Wide Web Consortium« oder normalerweise einfach als »W3C« bezeichneten Gruppe fest gelegt werden. Umfassende Informationen über das W3C finden Sie auf der Website dieser Gruppe unter http://www.w3.org/.

Da häufig neue Empfehlungen für mit XML in Zusammenhang stehende Technologien veröffentlicht werden, und sich die Software so schnell weiterentwickelt, dass die aktuellen W3C-Empfehlungen ständig überholt und daher neue Versionen der Standards erforderlich werden, erweist sich diese einfache Antwort jedoch als schwieriger, als es auf den ersten Blick scheint.

Wir sehen die Sache so: Sofern Sie nicht von einem bestimmten Softwareentwicklungshintergrund kommen, besonders von einem solchen, der SGML verwendet, werden Sie wahrscheinlich einiges an konzentrierter Lernzeit investieren müssen, um alle Aspekte zu verstehen, die bei der Entwicklung von XML-Webanwendungen mit Java ein Rolle spielen. Es ist daher empfehlenswert, dass Sie sich am aktuellsten verfügbaren Standard orientieren. Seien Sie sich dabei allerdings bewusst, dass Sie gelegentlich versuchen werden, mithilfe von Software, die mehr Fehler aufweist als einige ältere, »stabilere« Versionen, etwas über XML zu lernen.

Der Vorteil dieser Vorgehensweise besteht darin, dass Sie, wenn Sie das zum Abschluss eines Projekts erforderliche Wissen erworben haben, bezüglich der Standards auf dem neuesten Stand sind. Hierdurch wird es weniger wahrscheinlich, dass Sie es Microsoft mit seiner XML-Unterstützung für IE5.0 gleichtun werden. Mit anderen Worten: Es ist weniger wahrscheinlich, dass sich zu einem späteren Zeitpunkt herausstellt, dass Sie etwas verwenden, das nur in einem »Dialekt« einer mit XML in Zusammenhang stehenden Technologie definiert wurde.

1.5.5 Von uns verwendete XML-Technologien

XML und die damit in Zusammenhang stehenden Technologien verfügen über aufregende Möglichkeiten, die Entwicklung des Internets voranzutreiben. Dies hat zu zahlreichen interessanten Projekten geführt, von denen jedes versucht, dieses Potenzial zu realisieren. Einige dieser Projekte liegen bereits als Produkte vor, die angeblich zum Einsatz bereit sind, während es sich bei anderen um rein experimentelle Projekte handelt. Die uns hier interessierenden Produkte sind diejenigen, die kostenlos verwendet oder zumindest getestet werden können. Die von uns als Entwickler zu beantwortende Frage lautet: Welche XML-relevante Software sollten wir verwenden, um das Erstellen von Webapplikationen zu lernen?

Uns stellt sich die Sache folgendermaßen dar: Wenn Sie unseren Rat befolgen und den zahlreichen Links nachgehen, die wir Ihnen nennen, werden Sie während Ihrer Reisen im Cyberspace mehreren alternativen Softwareprogrammen begegnen, die zur Verwendung mit XML empfohlen werden können. Dies wird besonders dann der Fall sein, wenn Sie die Mailinglisten abonnieren und die Diskussionen, in denen Entwickler die Versuche kommentieren, mit verfügbaren Inhalten lauffähige Anwendungen zu erstellen, mitverfolgen bzw. wenn Sie sich an diesen Diskussionen beteiligen. Dabei ist es gut möglich, dass Sie in Erfahrung bringen, welche Komponenten Sie später in Ihren eigenen Projekten vorzugsweise verwenden möchten. Wir würden uns sehr freuen, wenn Ihnen dieses Buch hilft, genau über diese Komponenten mehr Informationen zu erhalten.

Das Apache-XML-Projekt: Xerces und Xalan

Wir sind jedoch der Meinung, dass dieses Buch komplex genug ausfallen wird, auch wenn wir nicht versuchen werden, mehr als eine Komponente jedes Typs der erforderlichen mit XML in Zusammenhang stehenden Komponenten zu besprechen. Außerdem suchen wir nach Software, die für Lernzwecke kostenlos zur Verfügung steht. Wir haben uns daher entschieden, für unsere Beispiele das XML-Projekt der Apache Software Foundation, insbesondere Xerces und Xalan, zu verwenden.

Weitere Informationen über die Apache Software Foundation, das Apache-XML-Projekt und die Ursprünge dieses Projekts finden Sie unter folgenden Links:

```
http://www.apache.org
http://xml.apache.org
http://xml.apache.org/pr/0001.txt
```

Warum gerade diese Tools? Bedenken Sie, dass sowohl Xerces als auch Xalan teilweise auf Quellcode basieren, der von IBM, Sun und anderen Unternehmen sowie von einzelnen Entwicklern mit reicher XML-Erfahrung der Apache Software Foundation gestiftet wurde. Offensichtlich haben sie sich entschieden, von den Vorteilen des »Open-Source«-Entwicklungsmodells (desselben Modells, das Linux und Apache Server so erfolgreich gemacht hat) zu profitieren, um ihre eigene XML-relevante Codebasis zu verbessern, zu entwickeln und zu testen. Sie haben sich ebenfalls entschieden, dass dies den besten Weg darstellt, eine Referenz-Codebasis für die sich entwickelnden Standards zu erstellen (durch das W3C).

Als Beleg dieser Tatsache lässt sich anführen, dass der aktuelle IBM XML-Parser eigentlich lediglich eine »Verpackung« für den Xerces XML-Parser darstellt, um die Kompatibilität mit der früheren Software von IBM zu gewährleisten. Sun war ebenfalls sehr großzügig bei der Bereitstellung von Quellcode für das Apache-

XML-Projekt und profitiert sicherlich von demselben weltweiten Entwicklerpotenzial, das IBM mit seiner »Open-Source«-Bewegung erschlossen hat. Bei unserer Entscheidung, Xerces und Xalan als unseren XML-Parser bzw. XSLT-Prozessor zu verwenden, befinden wir uns also in bester (großer?) Gesellschaft.

Auf eines können Sie sich mit Sicherheit verlassen: Beim gegenwärtigen Tempo der Entwicklung in der XML-Welt wird es, wenn Sie diese Zeilen lesen, bereits bessere als die von uns derzeit verwendeten Versionen von Xerces und Xalan geben. Sie können sich sicher leicht vorstellen, wie schwierig der Versuch ist, den Inhalt dieses Buchs detailliert genug zu halten, um relevant genug zu sein, jedoch gleichzeitig allgemein genug, um auch dann noch anwendbar zu sein, wenn neuere Versionen von Xerces und Xalan erschienen sind.

Um dieses Ziel zu erreichen, müssen wir, wie bereits erwähnt, davon ausgehen, dass ein Großteil Ihrer Lernbemühungen darin bestehen wird, den von uns angegebenen Web-Links zu folgen. Nur dann werden Ihre Lernmaterialien dynamisch genug sein, um dem neuesten Stand gerecht zu werden. Was Sie aus unserer Erfahrung lernen können, wird möglicherweise hauptsächlich dieses sein: Dass Sie zur Entwicklung lauffähiger Software eine ausdauernde Grundhaltung benötigen, und dass Sie eine Reihe geeigneter Ausgangspunkte benötigen. Wir werden uns in den nachfolgenden Kapiteln bemühen, diese beiden Tatsachen deutlich werden zu lassen.

Während Sie Ihre Kenntnisse über XML erweitern, werden von den Komponenten, die Sie zu verwenden versuchen, mit Sicherheit neue Versionen herausgegeben werden, und diese werden ihrerseits neuere Versionen anderer von Ihnen verwendeter Komponenten erforderlich machen. Sie werden versucht sein, den neueren Versionen keine Beachtung zu schenken. Nach unserer Erfahrung sollten Sie auf die neueren Komponenten jedoch so früh wie möglich umsteigen, da diese Versionsänderungen häufig mit einer deutlichen Verbesserung der Software und Dokumentation einhergehen. Der Versuch, mit früheren Versionen weiterzuarbeiten, die bereits länger getestet und besser bekannt sind, ist in zahlreichen Entwicklungssituationen sinnvoll, jedoch nicht im Zusammenhang mit auf XML bezogener Software. Die Entwicklung dieser Software umfasst einfach zu viele wichtige und grundlegende Aspekte, um diese Änderungen ignorieren zu können.

1.5.6 Unsere Wahl eines Webservers

Wie bereits weiter oben erwähnt, sind die drei am weitesten verbreiteten Webserver IIS, Netscape und Apache. Wenn es jedoch um die Entwicklung von Webapplikationen mit Java geht, gibt es einen zwingenden Grund, einen anderen Webserver in Betracht zu ziehen: Tomcat. Wie Xerces und Xalan, wird auch Tom-

cat unter der Apache Software Foundation im Rahmen des Jakarta-Projekts entwickelt. Weitere Informationen über das Jakarta-Projekt finden Sie unter:

`http://jakarta.apache.org.`

Warum werden wir Tomcat verwenden? Nicht deshalb, weil er ein besserer Webserver (HTTP-Server) als Apache oder IIS ist. Sein für Java-Entwickler ausschlaggebendes Leistungsmerkmal besteht darin, dass er ein Java-Servlet-Container ist und außerdem Java Server Pages (JSP) verarbeiten kann. Java-Code auf JSP-Seiten verleiht einem Webserver die Fähigkeit, einem Webbrowser dynamische Inhalte bereitzustellen – Inhalte, die nur zu dem Zeitpunkt fest gelegt sind, zu dem der Browser eine Anfrage an den Webserver richtet. Weitere Informationen über die JSP-Technologie können Sie aus dem Internet erhalten, indem Sie das offizielle Dokument hierzu unter der folgenden Adresse von der Sun-Website herunterladen:

`http://java.sun.com/products/jsp/whitepaper.html`

Wenn Sie die Vorteile der JSP-Technologie nutzen möchten, gibt es hierzu außer der Verwendung von Tomcat noch alternative Wege. Wenn Sie beispielsweise Apache Server einsetzen, könnten Sie Jserv dazu verwenden, um JSP-Seiten-Anforderungen zu verarbeiten. Weitere Informationen über JServ finden Sie unter http://java.apache.org/. Wenn Sie Microsofts IIS-Webserver verwenden, könnten Sie JRun verwenden. Weitere Informationen zu JRun finden Sie auf der Allaire-Website unter der Adresse http://www.allaire.com/Products/JRun/. Darüber hinaus gibt es noch weitere Möglichkeiten unter den verfügbaren Webanwendungs-Framework-Produkten, die JSP verstehen.

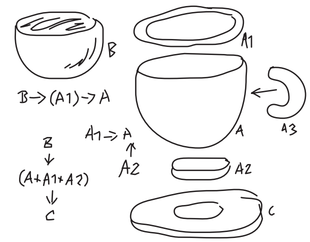
Obwohl Tomcat auch als Standalone-Webserver verwendet werden kann, wird erwartet, dass echte Webanwendungen den Server zusammen mit einem Apache Server verwenden. An späterer Stelle werden wir Ihnen den Weg zu den Informationen weisen, die Sie benötigten, um genau dies zu tun.

Die Intention des Jakarta-Projekts bei der Erstellung von Tomcat besteht jedoch darin, eine Referenzimplementierung für die Java-Servlet- und JSP-Technologien zur Verfügung zu stellen. Wenn Sie mehr über die neuesten Standards für diese beiden wichtigsten Java-Web-Technologien in Erfahrung bringen möchten, werden Sie mit Sicherheit mehr über Tomcat lernen wollen. Tomcat ist außerdem kostenlos verfügbare »Open-Source«-Software und passt damit in das Zielkonzept kostengünstiger Entwicklung, das wir hier verfolgen.

1.5.7 Plattformunabhängigkeit

Wenn Sie Ihre Webanwendungen auf der Programmiersprache Java aufbauen wollen, sind Sie sich wahrscheinlich bewusst, dass einer der Hauptvorteile dieser Sprache ihre Plattformunabhängigkeit ist. Sie werden es daher zu würdigen wissen, dass – mit Ausnahme des von uns verwendeten Browsers – sämtliche von uns ausgewählten Tools und Softwaremodule nicht auf ein bestimmtes Betriebssystem angewiesen sind. Apache Server, Tomcat, Suns Java Development Kit, Xalan und Xerces können sowohl auf Windows NT- als auch auf Unix-Systemen installiert werden.

Wir hoffen, dass die nächste Auflage dieses Buchs die Verwendung der von uns ausgewählten Webanwendungs-Tools auf Servern, auf denen Linux ausgeführt wird, detaillierter darstellen wird. Während Sie die Programmierung mit XML, XSLT, Java und JSP auf einem NT-Server lernen, können Sie jedoch zwischenzeitlich sicher sein, dass sich Ihre neu erworbenen Kenntnisse mühelos auf Betriebssysteme vom Unix-Typ übertragen lassen.



2 Vorbereitungen für die Java-Softwareentwicklung

In diesem Kapitel lernen Sie, wie Sie eine kostengünstige Java-Entwicklungsumgebung einrichten können. In dieser Umgebung können Sie unsere XML Java-Codebeispiele kompilieren, auf Fehler überprüfen und ausführen. Wir versuchen, den Java-Code möglichst einfach zu halten, da das Thema dieses Buchs XML, XSLT, Applets, Servlets und JSP ist.

2.1 Die Standardausgabe der Java™ 2-Plattform

Für sämtliche Java-Codebeispiele in der zweiten Hälfte dieses Buchs werden wir die Java™ 2-Plattform verwenden. Um die Beispiele nachvollziehen und dadurch eigene Programmierpraxis sammeln zu können, sollten Sie zunächst überprüfen, ob Sie auf Ihrem System Java 2-Quellcode kompilieren und ausführen können.

Es ist natürlich möglich, insbesondere da Sie sich zum Kauf dieses Buchs entschieden haben, dass Sie bereits mit einer Java 2-Entwicklungsumgebung arbeiten, vielleicht mit einem der zahlreich verfügbaren Produkte wie z. B. mit JBuilder™ der Inprise Corporation, Visual Age™ von IBM oder Visual Cafe 4™ von Symantec. In diesem Fall werden Sie es möglicherweise vorziehen, zum Kompilieren und Ausführen der Beispiele und zur Unterstützung Ihrer eigenen Bemühungen in der XML-Programmierung Ihre aktuelle Umgebung zu verwenden. Wenn dies für Sie zutrifft, können Sie die meisten Anweisungen dieses Kapitels überspringen.

2.1.1 Das Java™ 2 SDK

Für Leser, die eine Java 2-Entwicklungsumgebung einrichten müssen oder dieselbe Umgebung verwenden möchten, die uns beim Schreiben dieses Buchs zu Grunde lag, werden wir jetzt im Überblick darstellen, wie Sie das Java™ 2 SDK erhalten und installieren können. SDK ist das Akronym für »Software Development Kit« und die hier verwendete Software steht auf der Sun-Website zur Verfügung.

Um sie herunterzuladen, besuchen Sie http://java.sun.com, und folgen Sie den Produkt-Links zur Download-Seite für die Windows-Version des Produkts.

Versionen

Als wir die Arbeit an diesem Buch begannen, war Version 1.2.2–001 die neueste verfügbare Version der Standardausgabe des Java™ 2 SDK. Nachdem wir etwa die Hälfte des Buchs geschrieben hatten, wechselten wir zur Version 1.3.

Wir empfehlen Ihnen eine 1.3.x-Version zu verwenden, sofern Sie nicht einen Grund haben, eine 1.2.x-Version zu verwenden (beispielsweise weil Sie diese Version installiert haben und Sie keine 30-Megabyte-Dateien über eine langsame Modemverbindung herunterladen möchten). Wir werden uns bemühen, sämtlichen Java-Code in den Beispielen so generisch wie möglich zu halten. Wir hoffen, dass Sie bei der Durcharbeitung dieses Buchs entweder die Version 1.2.x oder 1.3.x verwenden können.

In den weiteren Kapiteln des Buchs gehen wir davon aus, dass auf Ihrem System Version 1.3 des Java™ 2 SDK installiert ist. Sollte dies nicht der Fall sein, müssen Sie die in unseren Anweisungen angegebenen Dateipfade entsprechend ändern.

Herunterladen des Java™ 2 SDK

Möglicherweise verfügen Sie bereits über das benötigte Java SDK und haben es auf Ihrem System installiert. Andernfalls können Sie beide Versionen herunterladen, indem Sie sich zunächst bei der Java Developer Connection registrieren lassen. Erläuterungen, wie Sie hierbei vorzugehen haben, werden Sie finden, wenn Sie sich mit der Adresse http://java.sun.com/products/jdk verbinden.

Sie haben die Wahl, entweder die gesamte Software auf einmal (etwa 30 Megabyte) oder aber über zwanzig kleinere Dateien herunterzuladen, die anschließend wieder zusammengesetzt werden müssen (jede Datei passt auf eine Diskette). Wenn Ihre Download-Sitzung nicht unterbrechungsgeschützt ist, schlagen wir Ihnen die letztere Möglichkeit vor. Es ist äußerst frustrierend, 25 Megabyte eines umfangreichen Downloads abzuwarten, nur um dann die Verbindung zu verlieren!

Wenn Sie die herunterladbare Installationsdatei ausführen, wird ein Installationsprogramm gestartet, das sämtliche Dateien in einem Ordner expandiert. Wir haben den vorgegebenen Ordner C:\jdk1.3 verwendet. Wir empfehlen Ihnen nach Möglichkeit diesen Ordner ebenfalls zu verwenden, da Sie dem Rest des Buchs dann leichter folgen können.

Vorbereitung zur Verwendung des Java™ 2 SDK

Es gibt eine ungeheure Vielfalt von Informationen über das Java™ 2 SDK, sodass wir Sie bitten möchten, bei Fragen zu diesem Thema an anderer Stelle nach Antworten zu suchen. Laden Sie die Dokumentation der von Ihnen verwendeten Version herunter, oder lesen Sie sie online. Insbesondere wenn Ihre Internetverbindung gebührenpflichtig ist, erscheint es vorteilhaft, die Dokumentation auf Ihrem System zu installieren.

Wir haben die Erfahrung gemacht, dass eine hervorragende Möglichkeit zur Einsicht der Dokumentation darin besteht, eine Version herunterzuladen, die vom HTML-Format in das einer Windows-Hilfedatei konvertiert wurde. Diese Version der Dokumentation stand auf folgender Website zur Verfügung:

```
http://erzatz.free.fr/jdk13.html
```

Die Datei, die Sie für Version 1.3 des SDK herunterladen müssen, hat den Namen »Jdk13b1.zip« oder einen ähnlichen Namen (dieser Name ist derjenige der Hilfe für Version 1.3beta1).

Für den Fall, dass Sie Ihre eigene Dokumentation in einem MS-Windows-Format verfügbar machen möchten, können Sie auf der »Ersatz«-Website umfassende Informationen über das Tool einsehen, das für diese Formatumwandlung verwendet wurde.

Einstellen von CLASSPATH

Wenn Sie nach Erläuterungen darüber suchen, welche Einstellung Sie für die CLASSPATH-Umgebung wählen sollten, können wir Ihnen folgendes ausgezeichnete Internetdokument empfehlen:

```
http://java.sun.com/products/jdk/1.3/docs/tooldocs/win32/
classpath.html
```

Eine bessere Informationsquelle bei Fragen zu diesem Thema ist uns nicht bekannt.

Überprüfen des SDK

Wenn Sie mit dem Java™ 2 SDK noch nicht vertraut sind, nehmen Sie sich etwas Zeit es kennen zu lernen. Seien Sie nicht überwältigt! Der Kit ist äußerst umfangreich, und Sie sollten sich freuen, dass Sie all diese Räder nicht noch einmal erfinden müssen!

Er enthält unter anderem die umfangreiche Datei »tools.jar«, die sich im Ordner »C:\jdk1.3\lib« befindet. Diese jar-Datei ist sehr wichtig, da sie den Java-Compiler enthält, den unsere Webanwendungen verwenden werden.

Beachten Sie ferner, dass der Kit einen Ordner mit dem Namen »C:\jdk1.3\jre« enthält. Wenn Sie bereits über Vorwissen zu Java verfügen, werden Sie erkennen, dass es sich hierbei um eine Java-Runtime-Umgebung handelt, die Ihnen die Ausführung von Java-Anwendungen erlaubt. Der Kit enthält im Ordner »C:\jdk1.3\jre\lib« noch eine weitere umfangreiche jar-Datei mit den Namen »rt.jar«. Diese Datei ist die Runtime-Bibliothek, die jede Java-Anwendung verwendet.

Die Java-Runtime-Umgebung, die zum Lieferumfang des Java™ 2 SDK gehört, ist nicht identisch mit derjenigen, die kostenlos verteilt werden darf. Sie dürfen sie nur während der Entwicklung verwenden. Auf die kostenlos verteilbare Java-Runtime-Umgebung werden wir in einem späteren Abschnitt dieses Kapitels näher eingehen.

Beachten Sie jedoch beim Erstellen einer Anwendung, die einen Compiler benötigt, dass dem Benutzer der Anwendung ein passender Java-Compiler für die Anwendung zur Verfügung stehen muss. Die Webanwendungen, die Sie in diesem Buch kennen lernen, setzen die Verfügbarkeit eines Java-Compilers voraus. Der Grund hierfür ist, dass JSP-Seiten, um einsetzbar zu sein, zu Java Servlets kompiliert werden müssen.

Dies bedeutet, dass die kostenlos verteilbare Java-Runtime-Umgebung zur Ausführung einer installierten Webanwendung, die JSP-Seiten kompilieren können muss, nicht ausreicht. Die rechtlichen und sonstigen Konsequenzen dieser Tatsache in Erfahrung zu bringen und einzuhalten, überlassen wir Ihnen. Für uns stellt sich die Sache so dar, dass der Server, auf dem Ihre Webanwendung installiert ist, über eine lizenzierte Version des Java-Compilers von Sun verfügen muss, zumindest solange, bis die Apache-XML- und Jakarta-Projekte andere Compiler verfügbar machen.

Einrichten eines nützlichen Befehlseingabefensters

Wir verwenden vorzugsweise die Befehlszeilenschnittstelle zum Java™ 2 SDK, da sie eine schnelle und einfache Methode zur Ausführung verschiedener Arbeitsschritte darstellt. Das NT-Befehlsfenster »Eingabeaufforderung« muss auf eine vom Standardmodus abweichende Weise eingerichtet werden, mit der die Arbeit nicht einfach ist. Wir möchten mit einem Fenster arbeiten, das über eine scrollbare Anzeige verfügt, da uns ansonsten zahlreiche Meldungen und Ausgaben entgehen, deren Text über die Fensterkoordinaten hinausgeht.

Suchen Sie im Windows NT-Explorer nach der Datei mit dem Namen »WinNT\System32\cmd.exe«. Erstellen Sie eine Verknüpfung zu dieser Datei. Kopieren Sie anschließend das neue Verknüpfungssymbol auf den Desktop.

Rechtsklicken Sie auf das Desktop-Symbol und wählen Sie im daraufhin angezeigten Popup-Menü die Option »Eigenschaften«. Gehen Sie dann zur Registerkarte »Layout«. Stellen Sie im Feld »Bildschirmpuffergröße« den Wert für »Breite« auf 128 und den für »Höhe« auf 512 ein. Stellen Sie im Feld »Fenstergröße« den Wert für »Breite« auf 78 und den für »Höhe« auf 32 ein.

Wenn Sie möchten, können Sie dieses Symbol in Ihren Startmenü-Ordner in einem Windows-NT-Profil verschieben und ihn in »Große, scrollbare Cmd.exe« umbenennen » (oder ihm einen beliebigen anderen Namen geben). Auf diese Weise steht Ihnen im Startmenü ein verwendbares NT-Eingabeaufforderungsfenster zur Verfügung.

Erweitern der SDK-Quelldateien

Wenn Sie auf Ihren Speichermedien über den dafür erforderlichen Platz verfügen, raten wir Ihnen, die Quellcodedateien zu entpacken, die Sie mit einer Java™ 2 SDK-Installation erhalten. Legen Sie hierzu einfach einen neuen Ordner für den Quellcode an. Wir haben unserem Ordner den Namen »C:\jdk1.3\src« gegeben. Öffnen Sie dann ein NT-Eingabeaufforderungsfenster, wobei Sie den neuen scrollbaren Anzeigetyp verwenden, den Sie zuvor erstellt haben. Verwenden Sie einen »jar«-Befehl, um zunächst den Inhalt einzusehen und ihn dann in Ihrem neuen »src«-Ordner zu dekomprimieren. Wenn Sie als Befehl zunächst einfach »jar« eingeben, werden Ihnen die Hilfeinformationen zu diesem Befehl angezeigt.

Mit folgendem Befehl können Sie den Inhalt der jar-Datei anzeigen:

```
jar -tvf src.jar
```

Wenn Sie den letzten Befehl ausprobiert haben, haben Sie eine gute Vorstellung davon bekommen, wie viel Code Sun Entwicklern zur Verfügung stellt. Seien Sie froh, dass Sie alle diese Java-Dateien nicht selbst schreiben müssen! Wenn Sie die erste Liste genauer einsehen möchten, bevor Sie die jar-Datei dekomprimieren, können Sie den folgenden Befehl eingeben und anschließend die dadurch erstellte Textdatei einsehen.

```
jar -tvf src.jar >srcjar.txt
```

Steht auf Ihrer Festplatte der hierfür erforderliche Platz zur Verfügung, können Sie den Quellcode in einer Dateihierarchie unterhalb des aktuellen Verzeichnisses erweitern. Geben Sie dazu einfach den folgenden Befehl ein:

```
jar -xvf src.jar
```

Testen der Java™ 2 SDK-Installation

Wenn Sie die Version 1.3 des Java-Entwicklungs-Kits installiert haben und anschließend überprüfen möchten, ob sie ordnungsgemäß funktioniert, führen Sie den folgenden Test durch. Öffnen Sie ein NT-Befehlseingabefenster. Ändern Sie mit einem Befehl wie dem folgenden den aktuellen Ordner:

```
cd c:\jdk1.3\demo\jfcylepad
```

Geben Sie nun den folgenden Befehl ein, um die Java-Demo-Anwendung »Stylepad« auszuführen:

```
java -jar Stylepad.jar
```

Je nach Ihrem System kann das eine Weile dauern, Sie sollten jedoch schließlich den »Stylepad« sehen, der eine Geschichte für Sie enthält. Testen Sie diese und andere Demo-Anwendungen, um einen Überblick über die Leistungsfähigkeit von Java zu bekommen.

Einstellen des Pfads in der Systemumgebung

Ein weiterer Arbeitsschritt, den Sie ausführen sollten, ist das Hinzufügen des Systempfads zu Ihrer NT-Systempfadvariablen. Sie können dies mithilfe des System-Tools der Systemsteuerung erreichen. Machen Sie die Registerkarte für die Umgebungseinstellungen ausfindig, und bearbeiten Sie die Pfadvariable. Sie müssen der Variablen ein Semikolon und anschließend »c:\jdk1.3\bin« hinzufügen (oder was immer die Entsprechung auf Ihrem System ist).

Unsere Pfadumgebungsvariable sieht folgendermaßen aus:

```
Path=c:\jdk1.3\bin;c:\winnt\system32;c:\winnt
```

Einstellen anderer Umgebungsvariablen

Während Sie die Pfadvariable einstellen, können Sie auch gleich alle anderen Variablen einstellen, die definiert sein müssen, bevor wir mit der Entwicklungsarbeit beginnen können. Zwei dieser Variablen, JAVAC und JAVA_HOME, stehen mit der Java-Entwicklungsumgebung in Zusammenhang. Die anderen Variablen, ANT_HOME, TOMCAT_HOME und ELIXIR_HOME, werden für Software eingestellt, die wir später in diesem Buch einrichten werden.

Sie mögen natürlich Gründe haben, warum Sie die Einstellungen nicht permanent in Ihre Umgebung aufnehmen wollen. Erstellen Sie in diesem Fall eine Stapeldatei, die diese Einstellungen nur für einen NT-Eingabeaufforderungsprozess vornimmt.

Unabhängig davon, wie sie im Einzelnen vorgenommen werden, sind dies die Variablen, die wir auf unserem System definieren, sowie ihre jeweiligen Werte. Verwenden Sie diese Einstellungen als Beispiele für Ihre eigenen:

```
JAVAC = c:\jdk1.3\bin\javac.exe
JAVA_HOME = c:\jdk1.3
ANT_HOME = c:\ant
TOMCAT_HOME=c:\tomcat
ELIXIR_HOME=c:\elixir
```

2.1.2 Anzeigen der Sun-Demo-Applets

Verwenden von appletviewer.exe

Eines der Programme im Ordner »c:\jdk1.3\bin« trägt den Namen »appletviewer.exe«. Sie können es zum Testen von Java-Applets verwenden, ohne dafür einen Webbrowser zu benötigen. Sie können appletviewer jetzt testen, um sich zu vergewissern, dass Ihre Java-Installation korrekt funktioniert. Öffnen Sie hierzu ein NT-Befehlseingabefenster und verwenden Sie den Befehl »CD«, um »C:\jdk1.3\demo\applets\Animator« als aktuellen Ordner einzustellen. Geben Sie anschließend die folgende Befehlszeile ein:

```
c:\jdk1.3\demo\applets\ANIMATOR>appletviewer example1.html
```

Je nach Konfiguration, Geschwindigkeit und Auslastung Ihres Systems kann es eine Weile dauern. Das Animator Demo-Applet sollte jedoch schließlich im appletviewer-Fenster ausgeführt werden.

Verwendung des Internet Explorers 5.x

Wir werden in diesem Buch den Internet Explorer 5.5 als Webbrowser verwenden. Wenn auf Ihrem System irgendeine Version von Internet Explorer 5.x installiert ist, können Sie die Fähigkeit des Browsers zur Ausführung eines Java-Applets testen. Versuchen Sie dazu, die folgende Datei im Browser zu öffnen:

```
c:\jdk1.3\demo\applets\Animator\example1.html
```

Möglicherweise werden Sie bei einer neuen NT- und IE5.0-Installation wie wir feststellen, dass der Browser eine bestimmte erforderliche Java-Klasse nicht finden kann. Dass eine Klasse nicht gefunden werden kann, weist häufig auf ein Problem bezüglich der Variablen CLASSPATH hin. Obwohl uns Sun derzeit davon abrät, eine globale CLASSPATH-Variable in unserer Umgebung einzustellen, müssen wir dies dennoch tun, um dem Browser das Ausführen des Applets zu ermöglichen.

Verwenden Sie zum Hinzufügen oder Bearbeiten einer CLASSPATH-Variablen die Registerkarte »Umgebung« des System-Tools der NT-Systemsteuerung, damit die Variable eine Java-Runtime-jar-Datei enthält. Mit anderen Worten: In Ihrer Systemumgebung muss sich mindestens der folgende Pfad (oder seine Entsprechung auf Ihrem System) befinden:

```
CLASSPATH=c:\jdk1.3\jre\lib\rt.jar
```

Beachten Sie, dass Sie nach einer Änderung der Umgebung den Browser erneut starten müssen, damit er die Werte der neuen Umgebungsvariablen liest. Sie sollten jetzt in der Lage sein, sämtliche Sun-Demo-Applets mit Internet Explorer 5.x sowie mit dem Tool »appletviewer« von Sun auszuführen.

Vielleicht werden Sie wie wir feststellen, dass es nicht ratsam ist, die Variable CLASSPATH in der Systemumgebung zu definieren. Dadurch entsteht ein Konflikt zwischen der Definition der Variablen für Internet Explorer zur Ausführung von Applets sowie unserem Wunsch, die Variable nur zur Verwendung durch bestimmte Programme zu definieren.

An späterer Stelle werden wir die Verwendung des Plug-Ins von Sun besprechen. Mit diesem Plug-In können Sie in den meisten Webbrowsern Applets ausführen, einschließlich in Browsern solcher Hersteller und Versionen, die über keine integrierte Fähigkeit zur Ausführung von Applets verfügen.

2.1.3 Die Java™ 2-Runtime-Umgebung

Sie können sicher sein, dass der Benutzer Ihre mit dem Java 2 SDK entwickelten Softwareprodukte unter der Java 2-Runtime-Umgebung ausführen kann. Diese Runtime-Umgebung enthält lediglich die Java-Kernklassen und die Unterstützungsdateien, ohne sämtliche Entwicklungs-Tools.

Zum Lieferumfang des SDKs gehört eine eigene Kopie der Runtime-Umgebung, sodass Sie diese nicht separat herunterladen müssen, wenn Sie lediglich Software entwickeln.

Das SDK selbst dürfen Sie nicht weiterverteilen, sondern nur die Runtime-Umgebung. Dies ist einer der Gründe dafür, warum wir Ihnen erläutern, wie Sie das SDK von der Website http://java.sun.com herunterladen können, anstatt es Ihnen auf der in diesem Buch enthaltenen CD zur Verfügung zu stellen. Die Kosten für das Herunterladen der Software sollten zumindest teilweise durch die Vielfalt der sonstigen nützlichen Materialien, die Ihnen auf der Sun Java-Website dynamisch zur Verfügung gestellt werden, kompensiert werden.

Auf der Download-Seite befinden sich einige wichtige Links: zur README-Datei, den Versionshinweisen, den Features sowie den Lizenz- und Installationshinweisen. Installationshinweise für Version 1.2.2–001 haben wir in folgendem Webdokument gefunden:

```
http://java.sun.com/products/jdk/1.2/install-windows.html
```

Vielleicht sind zu dem Zeitpunkt, zu dem Sie diese Zeilen lesen, bereits ähnliche Hinweise für Version 1.3 vorhanden.

Diese Informationen können sich als sehr nützlich erweisen. Sie machen beispielsweise klar, dass der Name älterer Versionen des SDKs »Java Development Kit« (abgekürzt JDK) war und dass Sie noch immer Verweise auf den JDK finden (wir fanden einen solchen Verweis beispielsweise im Namen des Stammordners für die Installation).

Lesen Sie unbedingt die Binary Code-Lizenzvereinbarung von Sun Microsystems Inc., indem Sie den entsprechenden Links der Download-Seite folgen, bevor Sie das Java™ 2 SDK herunterladen und installieren. Als Lizenznehmer dieses Produkts sind Sie zur Einhaltung der Bedingungen dieser Lizenz verpflichtet, sodass Sie wissen sollten, welchen Bedingungen Sie zustimmen, wenn Sie diese Software verwenden.

2.1.4 Dokumentation für das Java™ 2 SDK

Während Sie das SDK herunterladen, sollten Sie sich überlegen, ob Sie auch die dazugehörende Dokumentation herunterladen möchten, die als separates Download zur Verfügung stehen sollte. Wir gehen hier davon aus, dass Sun diese Dokumentation für Version 1.3 ebenso verfügbar machen wird wie für Version 1.2.2–001, die wir unter folgender Webadresse gefunden haben:

```
http://java.sun.com/products/jdk/1.2/download-docs.html
```

Wenn Sie über eine günstige Internetverbindung verfügen, können Sie die Dokumentation natürlich auch über das Internet einsehen, sodass Sie sie nicht herunterladen müssen – was Ihnen einiges an Festplattenspeicher einsparen kann! Außerdem bietet dies den Vorteil, eine Version der Dokumentation verwenden zu können, die Ihnen eine Suchfunktion zur Verfügung stellt.

Wenn Sie sich entscheiden, die Dokumentation herunterzuladen, versuchen Sie für das SDK und die Dokumentation denselben Stammordner zu verwenden. Der vorgegebene Ordnername für die von uns heruntergeladene Version lautet »c:\jdk1.3\«. Die Dokumentation für das Java™ 2 SDK sollte daher im Ordner »c:\jdk1.3\docs« ihren Platz finden und das andere SDK-Download sollte in den Ordnern »c:\jdk1.3\bin, c:\jdk1.3\lib, etc.« installiert werden.

2.1.5 Sonstige Java-Entwicklungs-Tools

Wenn Sie beabsichtigen, umfangreiche Java-Projekte in Angriff zu nehmen, können Sie zahlreiche, kostenlos verfügbare Tools und Code-Bibliotheken finden (komplett mit Quellcode!), die Ihnen sowohl beim Erlernen als auch beim Implementieren der Software Zeit ersparen.

Bean-Entwicklungs-Kit

Sie könnten möglicherweise den Bean Development Kit von der Sun-Website herunterladen, um ihn zu testen. (Als ich dies tat, hieß er BDK1.1.) Der BDK erleichtert die Erstellung von Beans. Wenn Sie den BDK herunterladen, suchen Sie auf der Sun-Website auch die Datei beans.101.pdf und versuchen Sie, diese Datei herunterzuladen. Die Datei enthält eine hervorragende technische Beschreibung der Java Bean-Technologie.

Wie Sie bereits wissen, ist eine der Aufgaben, die Sie mithilfe von JSP ausführen können, die Verwendung der Java Bean-Technologie. Dies bietet Ihnen Vorteile, wenn Sie Ihre Klasseninstanzen serialisieren möchten. Darüber hinaus bietet es eine gute Möglichkeit, die verschiedenen Einsatzmöglichkeiten von JSP zu nutzen. Der BDK kann bei der Entwicklung von Webanwendungen wirklich hilfreich sein.

Erstellen von Projekten mit nmake.exe

Ein weiteres Tool, das Sie möglicherweise benötigen werden, ist ein Tool zum Kompilieren Ihrer Projekte. Wenn Sie schon immer lernen wollten, wie das JDK von der Befehlszeilenschnittstelle aus verwendet werden kann, besteht eine gute Gelegenheit zur Aneignung dieses Wissens darin, das Programm »nmake.exe« zur Erstellung Ihrer Java-Programme unter Windows NT zu nutzen. (Suchen Sie ansonsten nach dem GNU-Programm, das eine Alternative zu »nmake.exe« darstellt und verwenden Sie stattdessen dieses Programm!)

Die meisten Beispiel- oder Demo-Ordner, die Sie zusammen mit den von Ihnen heruntergeladenen Java-APIs finden, werden eine Art Programm zum Kompilieren sämtlicher benötigter Java-Dateien eines Projekts enthalten. Wie Sie jedoch noch sehen werden, verwenden wir einige andere Tools, die einen großen Teil der arbeitsaufwendigeren Aspekte der »Java-Programmierung für die Massen« umgehen.

2.2 Kompilieren von Java-Programmen

2.2.1 ElixirIDE

Die Verwendung einer IDE, d.h. einer integrierten Entwicklungsumgebung (»Integrated Development Environment«), hat einen großen Vorteil gegenüber dem Einsatz von Befehlszeilen-Tools. Einige der verfügbaren IDEs (insbesondere die kommerziellen) sind jedoch umfangreich, teuer und auf herkömmlichen Computern langsam. Darüber hinaus erfordern sie eine große Menge RAM. Ich kam zu dem Entschluss, dass es nicht ratsam sei, von meinen Lesern zu verlangen, dass sie eine dieser Umgebungen besitzen oder kaufen, sofern es sich dabei nicht um diejenige Umgebung handelt, mit der der Leser schließlich arbeiten möchten.

Nach unserer Erfahrung hat sich **ElixirIDE** von Elixir Technology Pte Ltd. als sehr nützlicher Java-Editor und als »Home Base« erwiesen. Es steht in der Version ElixirIDE-Lite zur Verfügung, die Sie kostenlos testen können, bevor Sie sie kaufen. Die Lite-Version ist auf zehn Dateien pro Java-Projekt beschränkt. Auf der zu diesem Buch gehörenden CD finden Sie Version 2.4.2 von ElixirIDE-Lite. Beachten Sie jedoch, dass Sie sehr wahrscheinlich eine neuere Version von der Elixir-Website unter der Adresse http://www.elixirtech.com herunterladen können.

2.2.2 Installieren von ElixirIDE-Lite

Wir gehen davon aus, dass sich auf Ihrem System die Installationsdateien für ElixirIDE sowie für die Dokumentation zu ElixirIDE befinden, die Sie entweder unserer CD entnommen oder von der Elixir-Website heruntergeladen haben. Beachten Sie, dass Sie weitere Informationen unter der Adresse http://www.elixirtech.com finden können. Ziehen Sie diese Informationen zurate, um die Anforderungen für die Ausführung von Elixir sowie die Voraussetzungen für den Erwerb des Vollprodukts in Erfahrung zu bringen. Wir selbst haben die folgenden Dateien heruntergeladen:

```
c:\download\elixir\ElixirIDE-2.4.2-Lite.zip
c:\download\elixir\ElixirIDE-2.4.0-Docs.zip
```

Entpacken Sie zunächst die Installationsdateien der Dokumentation (mithilfe von WinZip oder einem vergleichbaren Programm). Sehen Sie dann die Dokumentationsdateien durch. Lesen Sie zuerst die Datei ElixirIDE.html. Diese Datei enthält Informationen über die Voraussetzungen der Verwendung von ElixirIDE sowie Anweisungen zur Installation und Ausführung des Programms. Wir haben sämtliche Dateien in einem Ordner namens »C:\Elixir« entpackt. Wir empfehlen Ihnen für die Beispiele dieses Buchs ähnlich vorzugehen, damit Sie die gleiche Dateistruktur wie wir haben.

Nach dem Entpacken der beiden Dateien, haben Sie eine jar-Datei vor sich (ElixirIDE-2.4.2-Lite.jar), die sämtliche ElixirIDE-Klassen, eine Lizenzdatei, ein Änderungsprotokoll sowie die HTML-Dokumentation und Tutorial-Dateien enthält.

Öffnen Sie ein NT-Befehlseingabefenster, und geben Sie versuchsweise den Befehl »java« ein. Daraufhin sollten die Hilfeinformationen für das Java-Runtime-Programm angezeigt werden. Versuchen Sie auch, den Befehl »javac« einzugeben, durch den die Hilfeinformationen für den Java-Compiler angezeigt werden. Wenn diese beiden Java-Programme auf Ihrem System nicht gefunden werden, müssen Sie vor dem Aufruf von ElixirIDE sicherstellen, dass der Ordner, in dem sich die ausführbaren Dateien java.exe und javac.exe befinden, in der Umgebungsvariable PATH enthalten ist. Auf unserem System bedeutete dies, dass der vorhandenen PATH-Umgebungvariable mithilfe des System-Tools der NT-Systemsteuerung »c:\jdk1.3\bin« hinzugefügt werden musste.

2.2.3 Stapeldateien zum Start von ElixirIDE

Wir haben die Erfahrung gemacht, dass es die beste Methode zum Start von Elixir ist, die folgende Zeile in eine NT-Befehlsstapeldatei (in unserem Fall »c:\ElixirartElixir.bat«) einzutragen:

```
java -mx32m -jar c:\Elixir\ElixirIDE-2.4.2-Lite.jar
```

Testen Sie den neuen Befehl zum Starten von Elixir in einem NT-Befehlseingabefenster. Wenn alles ordnungsgemäß funktioniert, wird der Begrüßungsbildschirm der ElixirIDE angezeigt. Dies hat einige andere Änderungen zur Folge, die Sie im NT-Explorer überprüfen können. Beachten Sie, dass Elixir Ihrem NT-Benutzerordner einen Konfigurationsordner hinzugefügt hat.

Normalerweise befinden sich Ihre Benutzerordner in den NT-Profilen. Nehmen wir beispielsweise an, Sie verwenden Version 2.4.x. Wenn Sie sich als »Samuel« angemeldet und NT-Server unter Übernahme des standardmäßigen Installationspfads eingerichtet haben, sollten Sie einen ElixirIDE-Konfigurationsordner mit folgendem Namen finden:

```
c:\WINNT\Profiles\Samuel\.ElixirIDE\2.4
```

Manches ergab für uns zunächst keinen Sinn. Wir stellten später fest, dass wir manchmal an eine NT-Domäne angemeldet waren und dass wir zu anderen Zeiten ohne eine Netzwerkdomänen-Verbindung arbeiteten. In wieder anderen Situationen meldeten wir uns als lokaler NT-Server Administrator an. Auf diese Weise hatte NT unter dem Ordner WINNT drei verschiedene Profile für uns angelegt. Darüber hinaus verfügen wir jetzt über drei verschiedene ElixirIDE-Konfigurationsordner, einen in jedem Profil.

Vielleicht machen wir irgendetwas falsch! Wenn wir dies jedoch tun können, dann können es andere auch! Wir haben dieses besondere Problem gelöst, indem wir den Ordner mit den neuesten Dateien kopiert und alle anderen Ordner damit überschrieben haben (nachdem wir für alle Fälle Sicherungskopien davon angelegt hatten).

2.2.4 Elixir-Plug-In-Erweiterungen

Wenn Sie mit Version 2.4.x von ElixirIDE arbeiten, sollten Sie einen weiteren Ordner mit folgendem Namen finden:

```
c:\WINNT\Profiles\samuel\.ElixirIDE\2.4\ext
```

Beachten Sie, dass sich vor »ElixirIDE« in einem der Ordnernamen in diesem Pfad ein Punkt befindet. In den Ordner »ext« werden Sie Plug-In-jar-Erweiterungs-APIs für Elixir kopieren.

Besuchen Sie die Website http://www.elixirbase.com, über die Sie mit der weltweiten Gemeinschaft von Elixir-Anwendern in Kontakt treten können, und auf der Ihnen eine ganze Reihe nützlicher Plug-In-Erweiterungsmodule zur Verfügung stehen. Um mit diesen arbeiten zu können, entzippen Sie einfach die jar-Dateien aus den herunterladbaren Zip-Dateien in den Elixir-Erweiterungsordner (beispielsweise in den Ordner c:\WINNT\Profiles\Samuel\.ElixirIDE\2.4\ext). Wenn Sie Elixier erneut starten, lädt das Programm automatisch die Erweiterungen in diesem Ordner.

Wir empfehlen Ihnen die Verwendung von mindestens zwei dieser Plug-Ins, wenn nicht sogar von mehr. Sie finden einige von ihnen auf der beiliegenden CD.

Ein Klassenhierarchie-Inspektor

Die Plug-In-Datei namens **inspector.jar** erlaubt Ihnen die Überprüfung der Klassendateien in einer beliebigen Java-jar-Datei oder in einem Java-Paket. Sie können dem Elixir-Konfigurationsordner problemlos Datenbanken hinzufügen, sodass Sie die Klassen in den von Ihnen in Ihren Applikationen verwendeten Java-Paketen überprüfen können.

Beanshell, eine interaktive Java-Shell

Sie sollten sich unbedingt Beanshell besorgen, die unter der LPGL-Lizenz verteilt wird. Sie steht auf der bereits erwähnten Elixir-Basis-Website als Plug-In-Datei mit dem Namen **bsh.jar** zur Verfügung. Weitere Informationen über dieses äußerst nützliche Tool finden Sie auf der Beanshell-Website unter der Adresse http://www.beanshell.org.

Wir fanden es sehr lehrreich, Skripte für die Beanshell zu schreiben, die Java-Objekte erstellen. Wir können dann mit einer realen Instanz dieses Objekts interaktiv »spielen«, indem wir seine Eigenschaften und Methoden ausprobieren. Diese experimentelle Vorgehensweise ist häufig die beste zur Beantwortung Ihrer Fragen, besonders wenn diese etwa folgendermaßen klingen: »Was wird passieren, wenn ich das mit dieser Objektmethode versuche?« Wir werden an späterer Stelle von der Beanshell auf diese Weise Gebrauch machen.

Die Shell der Beanshell

Verwechseln Sie das Plug-In Beanshell nicht mit einer anderen Shell-Konsole, die Teil der ElixirIDE ist. Die Shell-Konsole gibt Ihnen Zugriff auf die System-Shell oder den Befehlsprozessor. Dies bedeutet, dass Sie innerhalb der ElixirIDE bleiben und dennoch GUI- oder textbasierte Programme ausführen können. Besonders wenn Ihr Computer für die zu einem bestimmten Zeitpunkt ausgeführten Programme gerade noch über ausreichend Arbeitsspeicher verfügt, können Sie durch die Verwendung dieser Shell Zeit sparen. Außerdem wird die Abfolge der von Ihnen innerhalb der Shell gegebenen Befehle in einen Befehlsverlauf aufgenommen.

2.2.5 Erstellen der bonForum-Projektordnerhierarchie

Wir beginnen jetzt mit unserem Beispiel eines Webanwendungsprojekts, dem wir den Namen »bonForum« geben. Wir werden es erst erläutern, und dann an späterer Stelle ausführlicher entwickeln. Jetzt beginnen wir zunächst damit, es als Projekt in ElixirIDE einzurichten.

> **Hinweis** Sie können sämtliche Ordner und Dateien für das Beispielprojekt dieses Buchs auf der Begleit-CD finden. Sie können sämtliche benötigten Dateien von dort auf Ihr System kopieren und dazu verwenden, diesem Buch zu folgen. Es könnte jedoch lehrreich sein, das Verfahren nachzuvollziehen, mit dem wir dieses Projekt eingerichtet haben. Darum stellen wir Ihnen zusätzlich zu den Dateien auf der CD auch diese Informationen bereit.

Wir werden hier zum Erstellen der für das bonForum-Projekt benötigten Ordner einen Shell-Prozess innerhalb von Elixir verwenden. Diese Ordner sind zum größten Teil mit denjenigen identisch, die für das Erstellen von Webanwendungen für den Apache Tomcat Server empfohlen werden. Wenn wir unser Projekt auf diese Weise einrichten, wird es leichter sein, unsere Webanwendung als ein WAR-Archiv auszubilden und Tomcat-Beispielen zu folgen, die diese Struktur verwenden.

Wählen Sie die Option »Shell« im Kombinationsfeld »Process«. Daraufhin sollte eine Befehlszeileneingabeaufforderung und darüber eine Konsole angezeigt werden. Geben Sie den Befehl »cmd« ein. Anschließend sollte ein Microsoft Windows NT(TM) Copyright-Hinweis gefolgt von einer Eingabeaufforderung, die dem Namen Ihres ELIXIR_HOME-Ordners entspricht, angezeigt werden. (Unter Windows 95/98 müssen Sie »command« statt »cmd« benutzen.)

> Wir setzen voraus, dass Sie Elixir in einem Ordner namens »C:\Elixir« installiert haben. Sollte dies nicht der Fall sein, verwenden Sie bitte in den im Verlauf dieses Buchs angegebenen Dateipfaden statt unseren Ihren eigenen ELIXIR_HOME-Pfad. Wir gehen des Weiteren davon aus, dass Sie die bonForum-Projektordner noch nicht angelegt haben.

Da Sie jetzt über eine NT-Befehlseingabezeile (bzw. MS-DOS-Eingabeaufforderung) verfügen, ist es sehr einfach, mit ihrer Hilfe die folgenden Aufgaben unter der Eingabeaufforderung auszuführen. (Beachten Sie, dass Sie eventuell bereits über einen Ordner mit dem Namen »C:\Elixir\Projects« verfügen. In diesem Fall erhalten Sie nach Eingabe des ersten Befehls eine Meldung, die Ihnen mitteilt, dass Sie mit dem nächsten Befehl fortfahren können.)

	Befehl	**Ergebnis des Befehls**
1	md Projects	C:\Elixir\Projects
2	cd Projects	Changed current folder and prompt
3	md bonForum	C:\Elixir\Projects\bonForum
4	cd bonForum	Changed current folder and prompt
5	md classes	C:\Elixir\Projects\bonForum\classes
6	md etc	C:\Elixir\Projects\bonForum\etc
7	md lib	C:\Elixir\Projects\bonForum\lib
8	md src	C:\Elixir\Projects\bonForum\src
9	md web	C:\Elixir\Projects\bonForum\web
10	cd web	Changed current folder and prompt
11	md docs	C:\Elixir\Projects\bonForum\web\docs
12	md images	C:\Elixir\Projects\bonForum\web\images
13	md jsp	C:\Elixir\Projects\bonForum\web\jsp
14	md servlets	C:\Elixir\Projects\bonForum\web\servlets

Tabelle 2.1 NT-Befehle zur Erstellung der Projektordnerhierarchie unter ELIXIR_HOME.

Wenn Sie zur Archivierung Ihrer Quellcode-Versionen CVS verwenden wollen, können Sie auch »md cvs« eingeben oder einen beliebigen anderen Ordner anlegen, der Ihnen in dem Projekt zur Verfügung stehen soll. Wenn Sie bisher nur den obigen Shell-Prozess verwendet haben, könnten Sie den Meldungsprozess in ElixirIDE erneut auswählen.

2.2.6 Erstellen von bonForum als Projekt in ElixirIDE

Bevor Sie dieses Projekt anlegen, stellen Sie sicher, dass Sie in Ihrem Dateisystem die oben angegebene Projektordnerhierarchie angelegt haben. Vergewissern Sie sich außerdem, dass Sie für das bonForum-Projekt die vorgeschlagenen Projekteinstellungen, ebenfalls wie oben beschrieben, gespeichert haben.

Wenn Sie für Ihre GUI-Komponenten ein bestimmtes Erscheinungsbild bevorzugen, wählen Sie in Elixir das Menü »Look and Feel«, und treffen Sie darin die Auswahl für Ihre Einstellungen.

Verwenden Sie zum Hinzufügen unseres Projekts zu Elixir den Menübefehl »Projekt Neu«. Wählen Sie den Ordner »ELIXIR_HOME\Projects\bonForum«, und erstellen Sie darin das neue Projekt. Geben Sie der ElixirIDE-Projektdatei den Namen »bonForum«. Geben Sie dazu in das Textfeld »Dateiname« »bonForum« ein.

In der Projektansicht in Elixir sollte daraufhin ein Ordnersymbol mit der Beschriftung »bonForum« zu sehen sein. **Herzlichen Glückwunsch!** Damit haben Sie die ersten Schritte eines Webanwendungsprojekts hinter sich.

Wenn Sie den Cursor einen Moment auf dem Projekt-Symbol ruhen lassen, zeigt Ihnen der daraufhin eingeblendete Hinweis, dass dieses Symbol eine Datei darstellt, die Elixir für Sie angelegt hat. Tatsächlich hat Elixir zwei Dateien mit Namen wie den folgenden für Sie angelegt:

```
c:\Elixir\Projects\bonForum\bonForum.project
c:\Elixir\Projects\bonForum.project.settings
```

2.2.7 Festlegen von bonForum-Projekteinstellungen in ElixirIDE

Da Sie jetzt ein Elixir-Projekt angelegt haben, ist es Zeit, einige Einstellungen hinzuzufügen. Wählen Sie im Menü »Project« die Option »Settings«. Als Nächstes sollte die Datei »bonForum.project.settings« zur Bearbeitung im Fenster des Editors angezeigt werden.

WorkRoot- und ClassRoot-Einstellungen

Nehmen Sie unter »General Settings« die Einstellungsänderungen vor, die erforderlich sind, damit anschließend »workroot« und »classroot« durch folgende Zeilen definiert werden:

```
WorkRoot=\\Elixir\\Projects\\bonForum\\src
ClassRoot=\\Elixir\\Projects\\bonForum\\classes
```

Pfadeinstellungen

Stellen Sie als Nächstes sicher, dass Ihre Path.Windows_NT-Einstellung (vorausgesetzt NT ist Ihr Betriebsystem) auf einen dem unseren entsprechenden Ausdruck eingestellt ist:

```
C:\\WinNT;C:\\jdk1.3\\bin;C:\\WinNT\\System32;
```

HelpPath-Einstellung

Die HelpPath.Windows_NT-Einstellung ist eine weitere, durch Semikolons getrennte Liste von Pfadnamen. Sie sind für die XML- und XSLT-Pakete bestimmt, die wir in einem anderen Kapitel besprechen werden. Wenn Sie zu späteren Versionen wechseln, müssen Sie die Versionsnummern entsprechend anpassen.

Als unsere HelpPath.Windows_NT-Einstellung haben wir die folgende Liste von Pfadnamen angegeben (alle in einer langen Zeile anstatt auf separate Zeilen verteilt, wie hier abgedruckt):

```
C:\\jdk1.3\\docs;
C:\\xerces-1_2_0\\docs\\apiDocs;
C:\\xalan_1_1\\docs\\apiDocs;
```

Über die Pfade zu den API-Dokumentationen für Xerces und Xalan haben wir von Elixir aus Zugriff auf die Dokumentation für unsere XML- und XSL-Tools, die wir in einem späteren Kapitel vorstellen werden. Wenn Sie andere Versionen von Xerces und Xalan verwenden, müssen Sie diese Pfade entsprechend anpassen.

SourcePath-Einstellung

Die SourcePath.Windows_NT-Einstellung erlaubt Ihnen das Einsehen von Quellcodedateien in Elixir. Wir haben unserer SourcePath.Windows_NT-Einstellung die folgenden Pfadnamen hinzugefügt (ebenfalls alle in einer langen Zeile anstatt auf separate Zeilen verteilt, wie hier abgedruckt):

```
C:\\jdk1.3\\src;
C:\\xerces-1_2_0\\src;
```

```
C:\\xerces-1_2_0\\samples;
C:\\xalan_1_1\\src;
C:\\xalan_1_1\\samples;
```

Debugger-Einstellungen

Wenn Sie mit JDK1.3 arbeiten (oder mit JDK1.2.x und das JPDA-Paket dafür von der Sun-Website herunterladen), sollten Sie die Vorteile der JPDA-Technologie nutzen, um sich das Debuggen Ihrer Projekte in ElixirIDE zu erleichtern. Aktivieren Sie hierzu den Debugger mit der folgenden Einstellung:

```
Debugger=YES
```

Sonstige Einstellungen

Es gibt noch weitere Einstellungen, wie beispielsweise diejenigen für die RCS-Versionierung, die alle von Elixir dokumentiert sind. Wir überlassen es Ihnen, diese zu verwenden oder zu ignorieren.

Beachten Sie, dass Sie einige dieser Einstellungen in bestimmten Fällen korrigieren müssen, z.B. wenn Sie zu einer neueren JDK-Version wechseln.

Speichern der Projekteinstellungen

Nachdem Sie an Ihrer bonForum.project.settings-Datei die obigen Änderung vorgenommen haben, stellen Sie sicher, dass sie gespeichert werden. Sie können hierzu entweder den Befehl »Save« im Menü »File« oder einen Tastaturbefehl verwenden. Schließen Sie die Datei mit dem Befehl »Close« aus dem Menü »File« oder mit dem Tastaturbefehl.

2.2.8 Standardmäßige Projekteinstellungen in Elixir

Möglicherweise werden Sie für Ihre Projekte in Elixir Einstellungen verwenden wollen, die einigen der oben genannten Einstellungen ähnlich sind. In diesem Fall können Sie die standardmäßigen Projekteinstellungen in einer anderen Konfigurationsdatei namens »default.project.settings« ändern. Sie sollten vielleicht jedoch zunächst in Erfahrung bringen, was in Ihrer besonderen Situation funktioniert, und erst dann die standardmäßigen Einstellungen für alle neuen Projekte bearbeiten.

Die Datei »default.project.settings« befindet sich im Konfigurationsordner des NT-Benutzerordners. Seien Sie vorsichtig! Man kann diese Datei mit dem Befehl »Project Open« als Projekt öffnen (Sie müssen dazu zunächst den Dateityp auf »Alle Dateien« geändert haben). Wenn Ihnen dieser Fehler unterläuft (wie es uns

eines späten Abends passierte), werden Sie in Ihrem Konfigurationsordner eine Datei mit dem Namen »default.project.settings.settings« finden, und beim Verlassen von ElixirIDE werden Ihnen jede Menge Meldungen zu Java-Ausnahmefehlern angezeigt.

2.2.9 Hinzufügen einer neuen Java-Klasse zu einem Projekt

Zunächst werden wir nur eine Java-Klassen-Quelldatei zu unserem neuen Projekt erstellen, damit Sie eine Möglichkeit kennen lernen, wie dies erreicht werden kann. Später werden wir natürlich unseren neuen Projekt-ordnern noch zahlreiche weitere Dateien hinzufügen.

Erstellen einer Java-Klasse in ElixirIDE

Wählen Sie nun das Symbol des bonForum-Ordners in der Baumanzeige des Projekts aus, indem Sie darauf klicken. Wählen Sie den Menüpunkt »Script/Java/Add Class« aus. Fügen Sie eine neue Java-Klasse hinzu, indem Sie den folgenden Namen als »CLASSNAME«-Skriptparameter eingeben:

```
de.tarent.forum.bonForum
```

Im Meldungsprozess sollte daraufhin eine Meldung zu der soeben erstellten neuen Java-Quelldatei angezeigt werden. Sie werden feststellen, dass Sie jetzt über eine Ordnerhierarchie für den Java-Paketnamen verfügen, den Sie der Klasse gegeben haben. Elixir hat jeden einzelnen Ordner für das Java-Paket erstellt, beginnend im Ordner »src« unterhalb des bonForum Elixir-Projektordners »bonForum«.

Jeder neu erstellte Ordner enthält auch eine neue Datei und wird daher in der Projektbaumanzeige von ElixirIDE als Symbol angezeigt. Diese Dateien haben denselben Namen wie der Ordner, in dem sie sich befinden, sowie die Erweiterung »folder«.

Sie verfügen jetzt auch über eine Java-Quelldatei mit dem Grundgerüst des Codes für Ihre Klasse. Elixir kann zusammen mit CVS dazu verwendet werden, parallele Versionen sicher zu archivieren und bei Bedarf mehreren Entwicklern verfügbar zu machen. Wir werden diese Möglichkeiten hier nicht näher beschreiben. Weitere Informationen hierzu finden Sie auf der Elixir-Website.

Hinzufügen einer Datei zu einem Projekt in Elixir

Rechtsklicken Sie auf das »bonForum«-Symbol in der Projektbaumanzeige. In dem daraufhin angezeigten Kontextmenü sehen Sie die Option »New folder«. Wir könnten diese Option dazu verwenden, jeden einzelnen Ordner der von uns so-

eben erstellten Ordnerhierarchie des Java-Pakets hinzuzufügen. Anschließend könnten wir mit der Option »Add file« unsere neue Java-Klassen-Quelldatei hinzufügen.

Lassen Sie uns stattdessen die Vorteile einer schnelleren Lösung nutzen. Wählen Sie die Menüoption »Recursive add«. Wählen Sie im daraufhin angezeigten Dialogfeld den folgenden Ordner aus:

```
ELIXIR_HOME\Projects\bonForum\src\de
```

Wenn Sie jetzt auf die Schaltfläche »Add« klicken, sollte ein neues Symbol mit dem Namen »tarent« unter dem bonForum-Symbol im Fenster der Projektbaumanzeige erscheinen. Durch Klicken auf die entsprechenden Symbole können Sie den gesamten Projektordner und die Dateihierarchie vollständig öffnen. Die Farbe des Symbols bedeutet, dass diese Quelldatei kompiliert werden muss.

2.2.10 Kompilieren von bonForum.java in ElixirIDE

Rechtsklicken Sie auf das bonForum.java-Symbol in der Projektbaumanzeige von ElixirIDE. Im daraufhin angezeigten Kontextmenü sollten Sie einen Kompilierbefehl auswählen können. Nach einer Weile wird sich die Farbe des Symbols ändern und die Meldung »Done« erscheint.

Wenn Sie im Dateisystem nachsehen, sollten Sie eine neue bonForum.class-Datei in der korrekten Ordnerhierarchie für das Projektpaket finden. Mit anderen Worten: Sie sollten eine Datei mit einem dem folgenden ähnlichen Namen finden:

```
C:\Elixir\Projects\bonForum\classes\de\tarent\forum\bonForum.class
```

2.3 Ausführen von Java-Programmen

2.3.1 Stapeldateien zum Kompilieren und Ausführen

Eine einfache Möglichkeit zum Kompilieren und Ausführen von Java-Programmen besteht in der Verwendung einer Stapeldatei. Im Folgenden sehen Sie das Beispiel einer solchen Stapeldatei zur Ausführung eines fiktiven Java-Programms. Dieses Beispiel geht davon aus, dass Ihre Systempfadvariable den Ordner mit dem Programm »java.exe« enthält.

```
set OLD_CLASSPATH=%CLASSPATH%
set CLASSPATH=%CLASSPATH%;c:\xalan_1_1\xerces.jar;
set CLASSPATH=%CLASSPATH%;c:\xalan_1_1\xalan.jar;
java de.tarent.xml.sax.SAXParseTest %1
set CLASSPATH=%OLD_CLASSPATH%
set OLD_CLASSPATH=
```

Sie können eine dieser Beispieldatei ähnliche Datei in einem Ordner mit einer kompilierten Java-Klassendatei ablegen und ihr den Namen RunSAXParseTest.bat oder einen beliebigen anderen Namen geben. Anschließend können Sie dann einen dem folgenden ähnlichen Befehl eingeben, um dieses Java-Programm mit einer XML-Datei als Argument auszuführen.

```
RunSAXParseTest "c:\xml\bonForum.xml"
```

Hier ist eine weitere Stapeldatei. Diese Datei kompiliert das Java-Servlet »servletToJSP«. Sie können diese Datei in dem Ordner mit der Quelldatei »servletToJsp.java« ablegen. Diese Stapeldatei setzt voraus, dass Ihre Systempfadvariable den Ordner mit dem Java-Compiler »javac.exe« enthält.

```
set OLD_CLASSPATH=%CLASSPATH%
set CLASSPATH=%CLASSPATH%;c:\xalan_1_1\xerces.jar;
set CLASSPATH=%CLASSPATH%;c:\xalan_1_1\xalan.jar;
javac servletToJsp.java
set CLASSPATH=%OLD_CLASSPATH%
set OLD_CLASSPATH=
```

2.3.2 Ausführen von Java-Programmen von ElixirIDE aus

Wählen Sie die Option »Show classpath« aus dem »Script«-Menü von ElixirIDE aus. Sie werden feststellen, dass ElixirIDE den »ClassRoot«-Wert aus der Projekteinstellungsdatei dem Wert der CLASSPATH-Umgebungsvariablen hinzugefügt hat, der beim Starten von ElixirIDE gültig war. In der Anzeige des Meldungsprozesses werden Sie eine Zeile sehen, die etwa so beginnt:

```
CLASSPATH=\Elixir\Projects\bonForum\classes;
```

Der Rest der Zeile entspricht dem CLASSPATH-Wert, der beim Start von ElixirIDE gültig war, einschließlich seiner jar-Datei.

Sie können Ihre kompilierten Programme sehr einfach von der ElixirIDE aus ausführen (vorausgesetzt ist hierbei natürlich, dass sie lauffähig sind). Wählen Sie hierzu einfach die Option »Execute« aus dem Kontextmenü, das angezeigt wird, wenn Sie auf das Projektsymbol rechtsklicken.

Wenn das Symbol in dem Kontextmenü nicht angezeigt wird, überprüfen Sie, ob die Klasse, die Sie auszuführen versuchen, eine »main«-Methode enthält. Die von uns weiter oben erstellte Datei »bonForum.java« enthielt beispielsweise keine solche Methode. Doppelklicken Sie auf ihrem Symbol im Projektfenster und bearbeiten Sie sie so, dass sie wie im folgenden Beispiel über eine »main«-Methode verfügt:

```
public class bonForum
{
   public static void main (String[] args) {
      System.out.println("Hello, World!");
   }
}
```

Beachten Sie, wenn Sie diese Änderungen gespeichert haben, dass das Symbol nun anstatt in Schwarz wieder in Gelb angezeigt wird. Dies weist darauf hin, dass die Quelldatei neuer als die kompilierte Klassendatei ist. Kompilieren Sie bonForum.java erneut. Sie sollten die Klasse jetzt ausführen können. ElixirIDE erstellt für Sie einen neuen Prozess mit dem Namen »de.tarent.forum.bonForum«. Die Ausgabe des Programms wird an sein Anzeigefenster geleitet. Dort sollten Sie Folgendes sehen:

```
java "de.tarent.forum.bonForum"
Hello, World!
```

Als wir dies ausprobierten, waren wir ziemlich überrascht, »Hello!« statt »Hello, World!« zu lesen. Dann erinnerten wir uns jedoch daran, dass wir bei früheren Experimenten eine de.tarent.forum.bonForum-Klassendatei unter dem Ordner »jdk1.3\classes« abgelegt hatten. Java Virtual Machine fand und führte diese Datei anstelle der von uns neu kompilierten Datei aus. Wenn wir in dieser früheren Klasse »Hello, World!« statt »Hello!« verwendet hätten, würden wir nie bemerkt haben, dass wir nicht unsere neue Klasse, sondern stattdessen eine alte ausgeführt hatten!

Klassenpfade können eine problematische Angelegenheit sein, wie dieses Beispiel zeigt. Wir möchten Ihnen raten, auf der Website java.sun.com nach einem Dokument namens »classpath.html« zu suchen. Sie werden es wahrscheinlich ebenso hilfreich finden wie wir.

2.4 Debuggen von Java-Programmen

Sie sollten das Benutzerhandbuch für ElixirIDE einsehen können, indem Sie in Ihrem Browser die folgende (oder die äquivalente) Datei öffnen:

```
C:\Elixir\IDEManual\ElixirIDEManual.html
```

In diesem Handbuch finden Sie Anweisungen für das Debuggen Ihrer Java-Programme in ElixirIDE.

ElixirIDE kann den JPDA-Debugger von Sun verwenden. Dieser Debugger gehört zum Lieferumfang von JDK1.3 (in tools.jar). Wenn Sie stattdessen jedoch mit

JDK1.2.2 arbeiten, müssen Sie die JPDA (jpda.jar) separat ausfindig machen und herunterladen.

Um den Debugger für ein Projekt verwenden zu können, müssen Sie die Debugger-Einstellung aktiviert haben. Außerdem müssen Sie sicherstellen, dass ElixirIDE die Datei finden kann. Um dies mit JDK1.3 zu erreichen, verwendeten wir die Stapeldatei »C:\ElixirartElixirDebug.bat«. Beachten Sie, dass diese Stapeldatei von einem Eingabeaufforderungsfenster aus ausgeführt werden muss.

```
rem THIS IS FOR USE WITH JDK1.3
rem This starts ElixirIDE-2.4.2-Lite WITH Sun JPDA debugger
set JPDAJAR= c:\jdk1.3\lib\tools.jar
set ELIXIRJAR= c:\Elixir\ElixirIDE-2.4.2-Lite.jar
set JASPERJAR= c:\jakarta-tomcat\lib\jasper.jar
set SERVLETJAR= c:\jakarta-tomcat\lib\servlet.jar
set XMLJAR= c:\jakarta-tomcat\lib\xml.jar
set XERCESJAR= c:\xalan_1_1\xerces.jar
set XALANJAR= c:\xalan_1_1\xalan.jar
set OLD_CLASSPATH=%CLASSPATH%
set CLASSPATH=%CLASSPATH%;%JPDAJAR%
set CLASSPATH=%CLASSPATH%;ELIXIRJAR%
set CLASSPATH=%CLASSPATH%;%JASPERJAR%
set CLASSPATH=%CLASSPATH%;%SERVLETJAR%
set CLASSPATH=%CLASSPATH%;%XMLJAR%
set CLASSPATH=%CLASSPATH%;%XERCESJAR%
set CLASSPATH=%CLASSPATH%;%XALANJAR%
java com.elixirtech.IDE
set CLASSPATH=%OLD_CLASSPATH%
set OLD_CLASSPATH=
```

Bei der Arbeit mit JDK1.2.2 haben wir die folgende Stapeldatei mit dem Namen »C:\ElixirartElixirDebug_jdk122.bat« erstellt, die ElixirIDE zusammen mit dem JPDA-Debugger startete. Bei der Datei handelt es sich um die selbe wie oben – mit Ausnahme der nachfolgend aufgeführten Zeilen, mit denen die PATH-Einstellung geändert wird, wobei eine andere jav-Datei für die JPDAJAR-Variable verwendet wird. Beachten Sie auch bei dieser Stapeldatei, dass sie von einem Eingabeaufforderungsfenster aus ausgeführt werden muss.

```
rem THIS COMMAND IS FOR USE WITH JDK1.2.2,
rem This starts ElixirIDE-2.4.2-Lite WITH Sun JPDA debugger
set PATH=%PATH%;c:\jpda\bin
set JPDAJAR= c:\jpda\lib\jpda.jar
rem CONTINUE HERE AS IN C:\Elixir\StartElixirDebug.bat!
```

2.5 Verwenden von Jars

2.5.1 Wie Java von einer Jar-Datei aus ausgeführt wird

Machen Sie folgendes Experiment, um zu sehen, wie praktisch jar-Dateien für die Implementierung von Software sind. Wechseln Sie mit dem unten angeführten (oder einem entsprechenden) Befehl in einem NT-Befehlseingabefenster vom aktuellen Verzeichnis zum folgenden Ordner:

```
Cd c:\jdk1.3\demo\jfc\SampleTree
```

Geben Sie in dasselbe Fenster nun den folgenden Befehle ein:

```
java -jar SampleTree.jar
```

Dieser Befehl wird ein Java-Programm von einer jar-Datei aus ausführen. Dies ist nur möglich, wenn die Manifest-Datei der jar-Datei richtig eingestellt ist. Falls im Manifest keine zu startende Klasse angegeben wurde, können Sie diese im Aufruf des Java-Interpreters hinter dem Namen der jar-Datei auch direkt angeben.

2.5.2 Elixir-jar-Dateien

Experimentieren Sie mit dem »Project Packager« im »Project«-Menü von ElixirIDE. Es ist sehr einfach, die Java-Klassen Ihres Projekts in eine jar-Datei in einem beliebigen Verzeichnis zu komprimieren. Sie können von dieser Möglichkeit Gebrauch machen, um die Klassen Ihrer Webanwendung in den Webapplikationsordnern von Tomcat einzusetzen. Weitere Informationen über den Project Packager finden Sie im ElixirIDE-Benutzerhandbuch.